Сложность булевых функций

Сложность булевых функций — метрическая характеристика булевых функций, с содержательной точки зрения выражающая трудность их вычисления; в частности, такой характеристикой (мерой сложности) может быть наименьшее число шагов, достаточное для вычисления данной функции в некотором классе алгоритмов, число функциональных элементов, достаточное для построения схемы, реализующей эту функцию, другие подобные величины.

Булевы функции и логические функциональные элементы. Естественной математической моделью простейших (однотактных) электронных схем являются логические функциональные элементы. *Функциональный элемент* — это абстрактное устройство с одним или несколькими входами, на которые подается информация, и с одним или несколькими выходами, информация с которых подается на входы других элементов или на выход всего большого устройства, содержащего этот элемент. Упомянутая информация содержится в состояниях входов или выходов элемента. Этих состояний у каждого полюса элемента (полюс — это совокупное название его входов и выходов) два, и они обозначаются нулем или единицей. Если состояния входов элемента определены (т. е. известны сопоставленные им значения 0 или 1), то значения на выходах элемента тоже однозначно определены, т. е. являются функциями значений входов. Эти функции имеют аргументы, которые могут быть равными только 0 или 1, и сами эти функции принимают только значения 0 или 1. Такие функции называются функциями алгебры логики, или короче, булевыми функциями (в честь английского математика 19 века Джорджа Буля). Говорят, что элемент реализует (на своих выходах) упомянутые булевы функции, однозначно определенные самим этим элементом.

Идея применения алгебры логики для моделирования работы электрических или электромеханических схем сейчас кажется естественной и совсем тривиальной, но впервые она была высказана только в начале прошлого века известным физиком Паулем Эренфестом, долгое время жившим в России. Развивать эту идею дальше он не стал, и ее забыли до конца тридцатых годов, когда она была одновременно и независимо опубликована в России физиком В. И. Шестаковым, в США инженером и математиком К. Шенноном и в Японии инженером А. Накасимой. По-видимому, первым был Шестаков, но его диссертация была опубликована спустя несколько лет после ее написания, и Шеннон опубликовал свою статью раньше.

Шеннон рассматривал задачу моделирования работы электромеханических устройств, построенных из реле, и предложил в качестве модели понятие контактной схемы. Если реле пропускало ток, то соответствующий ему контакт являлся замкнутым, и его состояние описывалось символом 1, а если нет, то контакт становился разомкнутым, и его состояние описывалось символом 0. С тех времен элементная база электронно-вычислительной техники полностью сменилась несколько раз, пройдя путь от электронных ламп ощутимого размера до миниатюрных транзисторов, представляющих из себя зоны кремниевого кристалла, размеры которых измеряются нанометрами, и которых на кристалле может располагаться несколько миллионов. Такие электронные схемы вначале их появления назывались интегральными схемами, потом большими интегральными схемами (английская аббревиатура LSI — large-scale integration), и наконец сверхбольшими интегральными схемами (английская аббревиатура VLSI).

Физика и химия происходящих в электронных схемах явлений также менялась вместе с изменением принципов их работы, но по-прежнему для их математического моделирования используется аппарат алгебры логики, только вместо контактных схем

(которые в определенных ситуациях также используются) применяются логические схемы из функциональных элементов. Разумеется, для изучения тонких аспектов поведения реальных электронных схем (например, для вычисления их различных числовых характеристик, скажем силы протекающих в них токов, потребляемой мощности, температуры различных зон) иногда приходится использовать сложные физические модели, использующие, в частности, дифференциальные уравнения, но для моделирования процессов обработки информации, происходящих в схеме, применятся алгебра логики. Конечно, с определенной точки зрения алгебро-логическая модель функционирования электронной схемы является грубым приближением (низкий потенциал обозначают нулем, высокий — единицей), но это приближение адекватно описывает работу схемы и без его применения ни анализ, ни синтез сколь либо сложных схем был бы невозможен.

Простейшим примером логического функционального элемента является элемент с одним входом и одним выходом, значение которого всегда противоположно значению входа. Это значит, что он реализует булеву функцию $f(x)$, определяемую равенствами $f(0) = 1$, $f(1) = 0$. Такую функцию называют логическим отрицанием, обычно обозначают символом $\neg x$ (или $\bar{x}$), а реализующий ее элемент называют инвертором.

Примерами элементов с двумя входами и одним выходом являются элементы, называемые конъюнктором и дизъюнктором. Первый из них реализует функцию, называемую в алгебре логики конъюнкцией. Она принимает единичное значение только тогда, когда оба ее аргумента равны единице (в остальных случаях она равна нулю). По существу она совпадает с обычной операцией умножения целых чисел, если ее применять только к числам, равным нулю или единице. Поэтому ее иногда называют логическим умножением и обозначают точкой, которую часто в записи формул опускают. Однако во многих случаях полезно подчеркнуть логическую природу этой функции и ее тесную связь с логической связкой, также называемой конъюнкцией, и тогда для ее обозначения используют символ &, который пишут между символами ее аргументов, например как $x\&y$, а не $\&(x, y)$, как следовало бы делать, придерживаясь функционального способа записи. В исчислении высказываний (разделе классической математической логики) символ конъюнкции, поставленный между двумя высказываниями, интерпретируется как союз «и», поэтому в англоязычных странах инженеры обозначают конъюнктор символом AND.

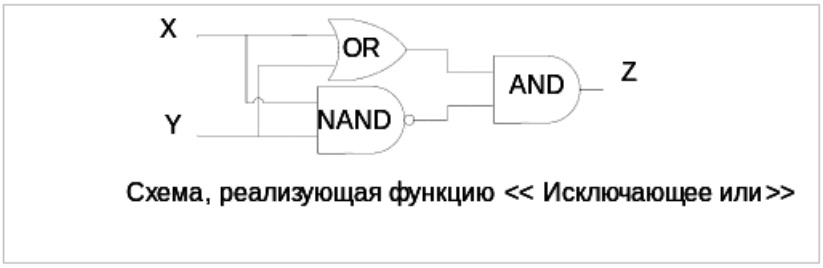
Другим примером двухвходового элемента является дизъюнктор, реализующий булеву функцию, называемую дизъюнкцией. Эта функция является двойственной к конъюнкции, и определяется двойственным образом, т. е. подобно конъюнкции, но с заменой 0 на 1 и наоборот. Более развернуто определение дизъюнкции можно дать следующим образом: дизъюнкция равна нулю тогда и только тогда, когда оба ее аргумента нули. Для обозначение дизъюнкции используется символ $\vee$, ее иногда называют логическим сложением, но нужно помнить, что в отличие от обычного сложения $1 \vee 1 = 1$, а не 2, а в англоязычных странах дизъюнктор обозначают символом OR, так как функция дизъюнкция связана с логической связкой, также называемой дизъюнкцией, которая интерпретируется как союз «или». В русском, как и в английском, союз «или» обычно понимается в разделительном смысле, который отличен, от того, как понимается этот союз в логике.

Но логическая связка, соответствующая разделительной дизъюнкции, в логике тоже, конечно, есть, и соответствующая ей булева функция принимает единичное значение тогда и только тогда, когда один из ее аргументов единица, а второй нуль. В алгебре логики ее часто обозначают символом $\oplus$ и называют суммой по модулю два, так как она действительно совпадает с этой алгебраической операцией. В англоязычных странах реализующий ее элемент обозначают символом XOR (eXclusive

OR).

Эту функцию можно выразить через упомянутые выше три функции, например следующим образом: $x \oplus y = (x\&\overline{y}) \vee (\overline{x}\&y)$. В алгебре логики часто приходится выражать одни булевы функции через другие, причем во многих случаях это можно сделать множеством способов, которые порождают множество логических тождеств. Например, справедливо тождество $x \oplus y = (x \vee y)\&\overline{(x\&y)}$.

Если у нас нет в распоряжении элемента XOR, но есть элементы OR, AND и NAND (так часто обозначают элемент, реализующий отрицание конъюнкции, используется также аналогичное обозначение NOR), то можно построить логическую схему из функциональных элементов, реализующую функцию XOR. Эта схема состоит из трех элементов, соединенных друг с другом, как показано на следующем рисунке.



Схема, реализующая функцию << Исключающее или >>

Схемы из функциональных элементов. Неформально говоря, схема из функциональных элементов есть произвольный способ соединения выходов некоторых элементов с входами других элементов, в котором не появляются замкнутые циклы (появление таких циклов в реальной однотактной схеме приведет к ее неустойчивой работе в некоторых ситуациях, а наличие циклов в логической схеме сделает невозможным корректное определение ее функционирования и в частности определение булевых функций, реализуемых выходами схемы). Схемы из функциональных элементов имеют такое длинное название, чтобы их отличать от других видов схем, также предназначенных для вычисления (или, как еще говорят, реализации) булевых функций, например, контактных схем. Так как другие виды схем далее не рассматриваются, то для краткости будем называть схемы из функциональных элементов просто схемами (в теоретическом программировании известно эквивалентное понятие, называемое неветвящейся программой). Многотактные схемы могут содержать и содержат циклы, но такие схемы реализуют не булевы операторы, состоящие из булевых функций, а конечно-автоматные отображения, имеющие память, в отличие от однотактных схем, не имеющих памяти. Однотактные схемы (схемы без памяти), построенные только из логических элементов, и не содержащие триггеров (флип-флопов), в инженерной практике часто называются комбинационными схемами, чтобы отличать их от автоматных схем. Комбинационные подсхемы часто занимают основную часть площади автоматных схем.

Пример такой схемы уже был приведен выше. Символы X и Y в этой схеме обозначают ее входы, теми же символами удобно обозначать булевы переменные (т. е. переменные, принимающие значения 0 или 1), значения которых определяют состояния этих входов в рассматриваемый момент. Символ Z обозначает ее выход. Если состояния входов схемы описываются значениями X и Y (иногда говорят, на входы поданы значения X и Y), то на выходе элемента OR появляется значение $X \vee Y$, на выходе элемента NAND — значение $\overline{X\&Y}$,а на выходе элемента AND (он же является выходом всей схемы) — значение $(X \vee X)\&\overline{(X\&Y)}$. Согласно указанному выше тождеству это значение совпадает с $X \oplus Y$. Говорят, что эта схема реализует функцию

$f(x, y) = x \oplus y$.

Сложностью (в англоязычной литературе — размером) схемы называется число составляющих ее элементов. В рассматриваемом примере сложность равна 3. В практических приложениях реальные схемы размещены на кремниевом кристалле, и под размером схемы естественно понимать ее площадь. Площадь схемы определяется суммарной площадью составляющих ее элементов и площадью, занимаемой проводами, соединяющими элементы. Задача минимизации площади схемы является важной прикладной задачей (в последнее время ее актуальность снизилась, так как современные технологии позволяют укладывать на кристалл схемы из огромного числа элементов).

Оценить долю площади, занимаемой проводами довольно сложно (она сильно зависит от используемого алгоритма укладки элементов схемы, называемого в англоязычной литературе placement, и от алгоритма разводки или трассировки проводов, называемого routing), и хотя у сложных схем она может быть довольно велика, грубо ее можно оценить как сумму площадей составляющих схему элементов. Поэтому иногда используется следующее обобщение понятия сложности схемы: сложность схемы — это сумма размеров (или весов) составляющих ее элементов.

В рассмотренном примере выход каждого элемента присоединяется к входу только одного другого элемента (говорят, что элементы схемы не имеют ветвлений) и схема имеет только один выход, и тем самым, реализует только одну функцию. Такие схемы часто называют формулами. Входы схемы в рассмотренном примере имеют ветвления (каждый из них присоединяется к входам двух элементов). Формулы, у которых и входы не имеют ветвлений, называют *бесповторными формулами*. Возможности таких формул для реализации булевых функций весьма ограничены.

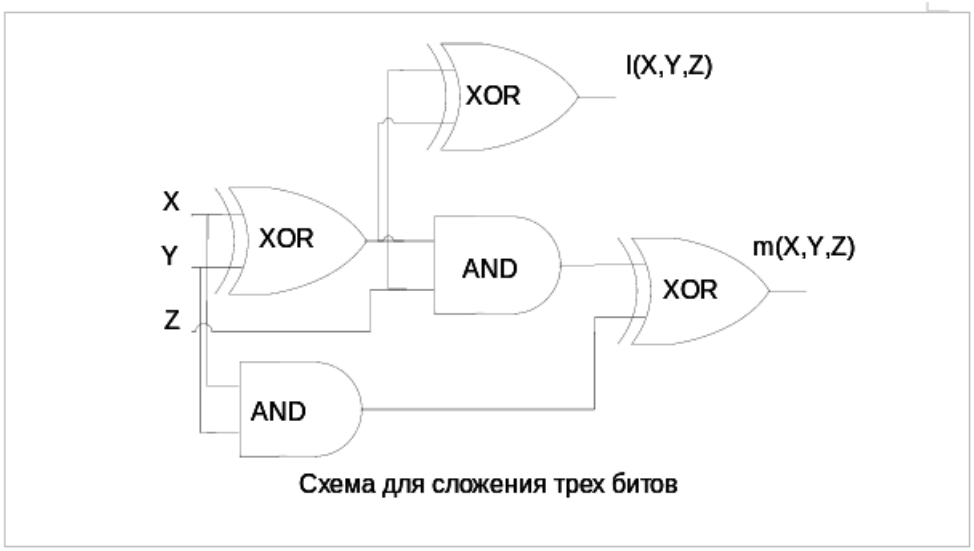
Рассмотренная схема построена из элементов OR, AND, NAND. Кружок при элементе NAND изображает инвертор (обозначаемый в англоязычной литературе NOT), но на самом деле технология такова, что элемент NAND составляется из отдельных частей, называемых в англоязычной литературе gate даже более просто, чем элемент AND, и имеет меньшую площадь и задержку при сравнимых электрических характеристиках. Набор различных элементов, использованных при построении схемы, называется ее базисом. На практике базис, который можно использовать при построении схем, обычно определен заранее и известен разработчику. В инженерной терминологии он называется технологической библиотекой. Современные библиотеки содержат сотни элементов (в англоязычной литературе их называют ячейками), элементы могут иметь три, четыре, а иногда и более входов, иногда два или несколько выходов, логически одинаковые элементы могут иметь разные модификации, имеющие разные значения площади, задержки и разные электрически характеристики. В теоретических исследованиях обычно используется базисы {OR, AND, NOT}, {OR, XOR, AND, NOT} или базис из всех двувходовых элементов, но иногда рассматривают и произвольные полные базисы из конечного числа элементов с произвольными неотрицательными весами этих элементов.

Одну и ту же функцию можно реализовать в разных базисах схемами разной сложности. Например, в базисе {OR, AND, NOT} функцию $x \oplus y$ можно реализовать схемой сложности 4 (для этого достаточно заменить в приведенной выше схеме отсутствующий в базисе элемент NAND на комбинацию элемента AND и инвертора). Можно доказать, что это схема минимальной сложности, реализующая функцию $x \oplus y$ в базисе {OR, AND, NOT}. Такие схемы называются минимальными схемами.

Задача построения для произвольной данной функции хотя бы одной минимальной схемы (минимальных схем может быть несколько) в данном базисе имеет важное прикладное значение, но она чрезвычайно сложна и далека от своего решения. Сложность минимальной схемы для данной функции называется сложностью самой

этой функции f и обозначается $L(f)$. Сложность функции может сильно зависеть от базиса, например функция $x \oplus y$ в базисе {OR, AND, XOR, NOT} очевидно имеет сложность 1. Поэтому в обозначение $L(f)$ снизу прибавляют в случае необходимости в виде индекса указание на используемый базис.

Ниже приводится более сложный пример схемы, в которой есть и ветвления элементов, и два выхода, которые не присоединяются ни к каким входам других элементов (но выход элемента, который присоединяется к входу другого элемента при желании тоже может быть объявлен выходом схемы).



В этой схеме входы обозначены соответствующими им символами булевых переменных X, Y, Z, а на двух ее выходах реализуются функции $l(x, y, z) = x \oplus y \oplus z$ — сумма трех переменных по модулю два (называемая также линейной функцией трех переменных или счетчиком нечетности с тремя переменными), и $m(x, y, z) = xy \vee xz \vee yz$, называемая функцией голосования комитета из трех человек, мажоритарной функцией или короче — *медианой*.

Первая из этих функций может быть определена следующим словесным описанием. Ее значение по аргументам x, y, z вычисляется как остаток от деления обычной суммы $x + y + z$ на 2. Равносильным образом можно определить эту функцию формулой $(x \oplus y) \oplus z$.

Медиану можно определить следующим словесным описанием. Она равна единице тогда и только тогда, когда не менее двух из ее аргументов единицы. Равносильным образом ее можно определить формулой $(xy \vee xz) \vee yz$ или чуть более короткой формулой $x(y \vee z) \vee yz$. Не так очевидно, что ту же функцию можно определить формулами $xy \oplus xz \oplus yz = x(y \oplus z) \oplus yz$.

Рассмотренная схема (ее в англоязычной литературе называют Full Adder и обозначают FA3) называется схемой сложения трех битов потому, что она действительно выполняет сложении трех одноразрядных чисел в двоичной системе по следующим формулам: $x + y + z = 2v + u$, $v = m(x, y, z) = xy \oplus xz \oplus yz$, $u = l(x, y, z) = x \oplus y \oplus z$. Для непосредственной проверки этих формул нужно перебрать все 8 значений булева трехмерного вектора (x, y, z) от $(0, 0, 0)$ до $(1, 1, 1)$. Но если учесть, что функции $l(x, y, z)$ и $m(x, y, z)$ симметрические, т. е. не зависят от порядка переменных, то перебор сокращается до четырех следующих вариантов: среди чисел x, y, z нет единиц, есть ровно одна единица, ровно две единицы, ровно три единицы. При этом следует заметить, что результат сложения трех битов в двоичной системе в рассматриваемых

случаях находится следующим образом: $0+0+0=0=(00)_2, 1+0+0=1=(01)_2$, $1+1+0=2=(10)_2, 1+1+1=3=(11)_2$.

Теория сложности булевых функций была развита в работах К. Шеннона, получившего в 1949 г. точную по порядку роста оценку сложности реализации булевых функций контактными схемами. В 1958 г. О. Б. Лупанов усилил этот результат и нашел асимптотически точную оценку сложности булевых функций в классе контактных схем. Несколько раньше О. Б. Лупанов нашел асимптотически точную оценку сложности реализации булевых функций в классе схем из функциональных элементов в произвольном конечном полном базисе булевых функций с положительными весами. В последующие годы теория сложности булевых функций превратилась в обширный раздел дискретной математики.

Развивая эти методы, сам О. Б. Лупанов, его ученики и последователи установили асимптотически точные оценки сложности булевых функций для большинства естественных классов схем и различных мер сложности (в частности, для формул, схем с ограниченным ветвлением, схем в базисах, содержащих элементы с нулевыми весами, схем в неполных базисах, схем в бесконечных базисах); были получены также аналогичные результаты о сложности реализации частичных булевых функций, функций многозначной логики и даже о сложности приближенной реализации обычных непрерывных функций действительного переменного.

Несмотря на то, что почти все булевы функции n переменных имеют экспоненциально высокую схемную сложность, не известно ни одного примера булевой функции (а точнее говоря, последовательности функций n переменных $\{f_n = f_n(x_1, \ldots, x_n)\}$), для которой, например в случае базиса {OR, AND, NOT}, была бы получена нелинейно растущая по n нижняя оценка сложности. Эффективное (в разумном смысле) построение такой функции вместе с доказательством того, что ее сложность растет быстрее любого полинома от n, решило бы в отрицательном смысле известную проблему NP-полноты (т. е. было бы доказано, что $P \neq NP$).

Рекомендуемая литература

1. О. Б. Лупанов. Асимптотические оценки сложности управляющих систем. М.: МГУ, 1984.

2. Конспект лекций О. Б. Лупанова по курсу «Введение в математическую логику» / Отв. ред. А. Б. Угольников. М.: Изд-во ЦПИ при мех.-матем. факультете МГУ им. М. В. Ломоносова, 2007.

3. С. В. Яблонский. Введение в дискретную математику. М.: Высшая школа, 2008.

4. С. В. Яблонский. Элементы математической кибернетики. М.: Высшая школа, 2007.

5. Дж. Сэвидж. Сложность вычислений. М.: Факториал, 1998.

6. Р. Г. Нигматуллин. Сложность булевых функций. М.: Наука, 1991.